

## Claims

What is claimed is:

1. A method for loading a kernel module, comprising:
  - loading a preload module into a kernel, wherein loading the preload module comprises creating a dynamic dependency list;
  - loading the kernel module comprising a static dependency list into the kernel;
  - triggering a hook when the static dependency list is reviewed;
  - obtaining module information associated with the kernel module using the hook;
  - searching for a dynamic dependency associated with the kernel module in the dynamic dependency list using the module information; and
  - updating the static dependency list with the dynamic dependency to obtain an updated static dependency list if the dynamic dependency associated with the kernel module is present in the dynamic dependency list.
2. The method of claim 1, further comprising:
  - defining a configuration file, wherein the configuration file is read by the preload module to create the dynamic dependency list.
3. The method of claim 1, further comprising:
  - updating the configuration file to obtain an updated configuration file;
  - unloading the preload module from the kernel;
  - re-loading the preload module into the kernel, wherein re-loading the preload module comprises reading the updated configuration file to generate an updated dynamic dependency list.
4. The method of claim 1, further comprising:
  - installing the hook using the preload module.
5. The method of claim 1, further comprising:

creating an interposer module, wherein the interposer module defines a new kernel symbol definition for a kernel symbol;  
defining a configuration file, wherein the configuration file indicates that the kernel module is dependent on the interposer module;  
loading the interposer module before the kernel module using the static dependency list;  
re-defining the kernel symbol using the new kernel symbol definition;  
loading the kernel module, wherein the kernel module comprises the kernel symbol; and  
resolving the kernel using the new kernel symbol definition.

6. The method of claim 5, wherein the interposer module comprises a reference count.
7. The method of claim 1, wherein the hook is installed in a kernel runtime loader.
8. The method of claim 1, wherein the kernel module comprises a flag indicating that a kernel symbol may have multiple definitions.
9. The method of claim 1, wherein updating the static dependency list comprises adding the dynamic dependency to the static dependency list.
10. A system comprising:
  - a kernel module having a static dependency list and a kernel symbol;
  - a modified kernel runtime loader configured to load the kernel module using the static dependency list; and
  - a preload module comprising a dynamic dependency list, wherein the preload module is configured to install a hook into the modified kernel runtime loader,wherein the hook is configured to update the static dependency list using the dynamic dependency list to obtain an updated static dependency list if a

dynamic dependency associated with the kernel module is present in the dynamic dependency list.

11. The system of claim 10, further comprising:
  - a configuration file configured to define a dynamic dependency,
  - wherein the configuration file is used to generate the dynamic dependency list.
12. The system of claim 11, wherein the configuration file is read by the preload module when the preload module is loaded.
13. The system of claim 11, wherein the preload module is unloaded and reloaded when the configuration file is modified.
14. The system of claim 10, further comprising:
  - an interposer module configured to provide a new kernel symbol definition for the kernel symbol, wherein the kernel module is dependent upon the interposer module.
15. The system of claim 14, further comprising:
  - a configuration file configured to define the dynamic dependency,
  - wherein the configuration file indicates that the kernel module is dependent upon the interposer module.
16. The system of claim 14, wherein the kernel symbol is resolved using the new kernel symbol definition.
17. The system of claim 14, wherein the interposer module comprises a reference count.
18. The system of claim 14, wherein the kernel module comprises a flag indicating that a kernel symbol may have multiple definitions.

19. The system of claim 10, wherein the preload module searches the dynamic dependency list using kernel module information to determine whether the dynamic dependency associated with the kernel module is present.
20. The system of claim 10, wherein updating the static dependency list comprises adding the dynamic dependency to the static dependency list.
21. A computer system for loading a kernel module, comprising:
  - a processor;
  - a memory;
  - a storage device;
  - a computer display; and
  - software instructions stored in the memory for enabling the computer system under control of the processor, to perform:
    - loading a preload module into a kernel, wherein loading the preload module comprises creating a dynamic dependency list;
    - loading the kernel module comprising a static dependency list into the kernel;
    - triggering a hook when the static dependency list is reviewed;
    - obtaining module information associated with the kernel module using the hook;
    - searching for a dynamic dependency associated with the kernel module in the dynamic dependency list using the module information; and
    - updating the static dependency list with the dynamic dependency to obtain an updated static dependency list, if the dynamic dependency associated with the kernel module is present in the dynamic dependency list.
22. A network system having a plurality of nodes, comprising:

a kernel module having a static dependency list and a kernel symbol;  
a modified kernel runtime loader configured to load the kernel module using the static dependency list; and  
a preload module comprising a dynamic dependency list, wherein the preload module is configured to install a hook into the modified kernel runtime loader,  
wherein the hook is configured to update the static dependency list using the dynamic dependency list to obtain an updated static dependency list if a dynamic dependency associated with the kernel module is present in the dynamic dependency list,  
wherein the kernel module executes on any node of the plurality of nodes,  
wherein the modified kernel module runtime loader executes on any node of the plurality of nodes,  
wherein the preload module executes on any node of the plurality of nodes.